1. Erläutern sie die Stufen der Zeichencodierung!

Zeichen müssen letztendlich in Bits und Bytes kodiert werden. Eine Zeichenkodierung gibt dazu eine Vorschrift an.
Das Kodierungsmodell stellt einen konzeptuellen Rahmen dar, innerhalb dessen man die Kodierung von potentiell einigen Milliarden Zeichen in Bitmuster strukturieren und so besser verstehen kann.

Die einzelnen Bestandteile sind:

1. ein abstrakter Zeichen
2. satz

2. eine Kodetabelle
3. ein Kodierungsformat
4. ein Kodierungsschema
5. eine Übertragungssyntax, die über so genannten .„glatten Text“ hinaus geht

Zwischen den verschiedenen Stufen bestehen Abbildungen.

Abstrakter Zeichensatz

Ein abstraktes Zeichen ist eine Informationseinheit, die zur Repräsentation, Organisation oder Kontrolle von Text dient, also beispielsweise Buchstaben, Ziffern, Interpunktionszeichen, Akzente, graphische Symbole, ideographische Zeichen, Leerzeichen, Tabulatoren, Zeilenweiterschaltung und -vorschub (engl. *Line Feed* und *Carriage Return)*, Kontrollkodes für Auswahl-Markierungen (engl. *Start of Selected Area* und *End of Selected Area* etc.).

Er hat keine Ordnung und besteht aus druckbaren sowie undruckbaren Zeichen. Beispiel „a“ oder Zeilenschaltung.

Designprinzipien für den Unicode-Standard.

1. Unicode kodiert Zeichen, nicht Glyphen
2. Unicode kodiert glatten Text, keine Formatinformationen.

Kodetabelle

Sie weist abstrakten Zeichen eine Kodeposition zu. Eine Kodeposition ist immer eine natürliche Zahl größer oder gleich null. Der Koderaum einer Zeichenkodierung ist immer ein Abschnitt der nichtnegativen natürlichen Zahlen, der die Null und alle Kodepositionen enthält. Der Koderaum kann jedoch auch Zahlen enthalten, die keine zulässigen Kodepositionen sind.

Unicode beinhaltet für jedes in seiner Kodetabelle kodierte Zeichen die folgenden Daten:
.• die Kodeposition des Zeichens
.• ein typisches Glyphenbild für das Zeichen
.• einen Namen
.• semantische Information

Von 10 Designprinzipien lassen sich fünf auf Ebene der Kodetabelle erklären:

1. Unifikation: Unicode repräsentiert Zeichen, die in verschiedenen Alphabeten vorkommen, aber vom Aussehen her ähnlich sind, nur einmal.
2. Konvertierbarkeit zwischen etablierten Standards: Eine eineindeutige Abbildung aus einem etablierten Standard nach Unicode soll ermöglicht werden. Zeichenpositionen aus einem einzelnen international verbreiteten Zeichensatz, die nach dem Designprinzip der Unifikation eigentlich miteinander identifiziert werden müssten, erhalten trotzdem getrennte Unicode-Kodepositionen.
3. Semantik: Unicode definiert semantische Eigenschaften für Zeichen.
4. Dynamische Komposition: Jedes Basiszeichen kann mit beliebig vielen Kombinationszeichen gleichzeitig gepaart werden.
5. Charakterisierung äquivalenter Kodierungen: Für die verschiedenen Kodierungen eines Basiszeichens mit Kombinationszeichen oder eines primären und eines aus Kompatibilitätsgründen in den Zeichensatz aufgenommenen Zeichens kann eine normalisierte Kodierung gefunden werden.

Kodierungsformat

Ein Kodierungsformat für eine Kodetabelle legt Bitrepräsentationen für die Kodeposition fest. Dazu bedient es sich einer Kodeeinheit, die in der Regel aus acht oder sechzehn Bits besteht. Ein Kodierungsformat bildet dann die Positionen eines Koderaums in Sequenzen von Kodeeinheiten und somit in Bitmuster ab. Wird jede Kodeposition einer Kodetabelle auf die gleiche Anzahl von Kodeeinheiten abgebildet, sprechen wir von einem Kodierungsformat fester Länge; andernfalls hat das Kodierungsformat variable Länge.

Das Unicode-Designprinzip der Effizienz lässt sich auf Ebene von Kodierungsformaten erläutern.

Kodierungsschema

Damit Daten über Netzwerke zuverlässig ausgetauscht werden können, müssen sie in eine Folge von Bytes serialisiert werden. Mithilfe eines Kodierungsformats ist es bis jetzt gelungen, einen Text als Folge von Kodierungseinheiten darzustellen. Ein Kodierungsschema hat nun die Aufgabe, zu einem Kodierungsformat zusätzlich festzulegen, wie die Kodierungseinheiten in Bytefolgen zu serialisieren sind.

Ist die Kodierungseinheit eines Kodierungsformats selbst schon acht Bits lang, so ist nichts mehr festzulegen und das Kodierungsschema ist mit dem Kodierungsformat identisch. Man kann also UTF8 sowohl als Kodierungsformat als auch als Kodierungsschema ansehen.
Ist die Kodierungseinheit ein Wyde, wie bei UTF16, so gibt es zwei Möglichkeiten der Serialisierung: *big endian* (das höherwertige Byte kommt zuerst) und *little endian*. Dementsprechend gibt es zu UTF16 zwei Kodierungsschemata, genannt UTF16-BE und UTF16-LE. Analoges gilt für UCS2 und UCS4.
Mit dem *Byte Order Mark* (BOM) an Position FEFF kann ein UTF16-repräsentierter Datenstrom signalisieren, welche der beiden Serialisierungen, *big endian* oder *little endian*, vorliegt.

Jenseits von glattem Text – eine Übertragunssyntax

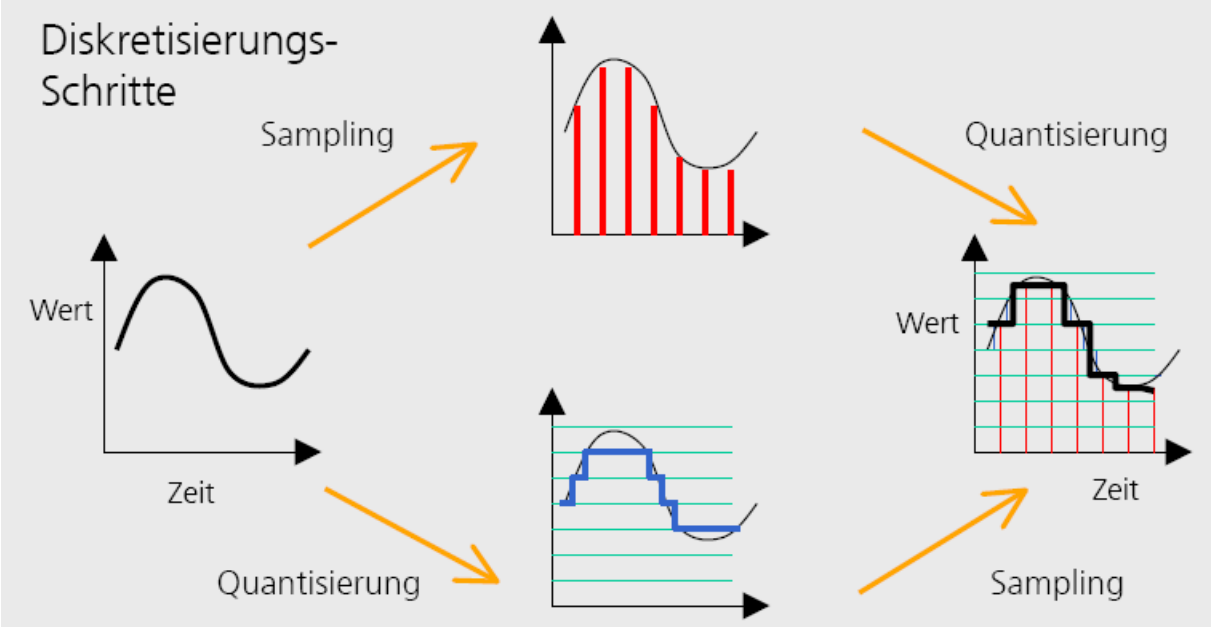
Die klassische Methode, Funktionszeichen in glattem Text unterzubringen, ist die Verwendung von *Escape*-Zeichen, die selbst nicht wörtlich als Bestandteil des Textes verstanden werden, sondern die Präsenz eines Zeichens signalisieren, das nicht direkt im Text vorkommen darf. Im Falle von XML bezeichnet die Formel &#<<Hexziffern>>; oder als &<<Dezimalziffern>>; im Inhaltstext das Unicode- Zeichen an Position <<Hexziffern>> bzw. an Position <<Dezimalziffern>>. Wir können also ein .„<." im Inhaltstext z. B. als < notieren und das zusätzliche Funktionszeichen .„&." durch &.

Die hier dargestellte Technik, Zeichen entweder durch sich selbst oder durch spezielle Zeichenfolgen wie &#<<Hexziffern>>; zu kodieren, löst insgesamt drei bekannte Probleme: Das erste Problem, wie sich in glattem Text Funktionszeichen an Stellen unterbringen lassen, an denen sie nicht im Klartext vorkommen dürfen, wurde bereits angesprochen. Dieselbe Technik ermöglicht es zweitens auch, Zeichen in einen Text einzufügen, die das verwendete Eingabewerkzeug nicht unterstützt. So kann man also beispielsweise in ein XML-Dokument mit einer amerikanischen Tastatur den dort nicht unterstützten Umlaut .„ä." als ä eingeben. Drittens ist es üblich, Paare aus den beiden ASCII-Steuerzeichen LF (engl. *linefeed*, *newline*) und CR (engl. *carriage return*) sowie jedes der beiden Zeichen für sich allein als ein- und dasselbe Zeilenendezeichen zu interpretieren. Auf diese Weise werden die beiden gängigen verschiedenen Konventionen, im Text ein Zeilenende zu signalisieren, vereinheitlicht. Werkzeuge, die die Texte verarbeiten, verfügen damit über eine plattformunabhängige Methode, Zeilen zu nummerieren.

Wie wird ein Audiosignal digitalisiert?

Erfolgt nach Drei-Stufen-Modell: Abtasten, Diskretisieren und Kodieren.

Beim *Sampling* und der *Quantisierung* findet eine Abtastung des zeit- und wertekontinuierlichen Signals einer Schallwelle in gewissen Intervallen statt. Die *Samplingrate* wird als Frequenz der Abtastung in der Einheit *Hertz* (abgekürzt *Hz)* angegeben. Alle Werte, die ggf. nicht zu den durch die Samplingrate festgelegten Zeitpunkten erfasst worden sind, werden dementsprechend verworfen. Nach dem Sampling ist das Signal zwar in eine endliche Anzahl von Werten zerlegt, diese können aber grundsätzlich immer noch unendlich in ihrer Ausprägung hinsichtlich des Wertebereichs sein. Deshalb folgt nun der zweite Schritt: die *Quantisierung.*
Für den Computer nicht ausreichend präzise repräsentierbare und damit korrekt verarbeitbare Werte .– wie etwa 1/3 oder Pi werden dabei zunächst auf den nächsten diskret repräsentierbaren Wert gerundet. Welcher Wert der am nächsten liegende Wert ist, wird durch die *Quantisierungsintervalle* festgelegt. Was bei der Abtastung die Samplingrate ist, ist beim Diskretisieren die Bitrate. Durch die Quantisierung entstehen Ungenauigkeiten, deren Effekt auch als *Quantisierungsrauschen* bekannt ist. Je höher die gewählte Bitrate zur Quantisierung gewählt wird, desto geringer ist die Wahrscheinlichkeit, dass störendes Quantisierungsrauschen auftritt.



Der letzte Schritt des Drei-Stufen-Modells ist die *Kodierung*. Hierbei werden die verschiedenen Quantisierungsintervalle mit binären Codewörtern gekennzeichnet. Das so entstandene digitalisierte Signal wird mitsamt dem *Quantisierungsfehler* übertragen, der jedoch bei geschickter Wahl der Bitrate nicht im Bereich des Hörbaren liegt.

Das hier beschriebene Verfahren zur Analog-/Digitalwandlung von Signalen wird auch *Waveform-Encoding* bzw. *PCM* (**P**ulse **C**ode **M**odulation) genannt. Es wird im weiteren Verlauf noch genauer erläutert.

Wie wird Audio komprimiert?

Verlustfrei durch Huffman-Kodierung

Sie gehört zum Entropy Coding Verfahren.
Dabei werden häufig vorkommen Signale mit wenigen Bit kodiert und selten vorkommende mit längeren Bitsequenzen. Zu diesem Zweck wird ein Binärbaum erstellt bei dem die Pfade zum seltensten verwendeten Zeichen am längsten sind. Man erreicht eine Komprimierung von 1:2.
Beispiel Huffman: Es wird zuerst festgestellt welche Zeichen wie häufig vorkommen. Dann wird ein Binärbaum erstellt und die Zeichen in aufsteigender (nach Vorkommen) Reihenfolge notiert. Danach werden die mit der geringsten Häufigkeit paarweise Verbunden. Die Summe ihrer Vorkommen wird im Elternknoten summiert. Danach wird immer weiter mit Nachbarn verbunden. Beim entstandenen Binärbaum wird dann bei jeder linken Abzweigung eine 0 und bei jeder rechten eine Eins erfasst. So dass das Zeichen welches am häufigsten vorkommt die Binärpresentation 0… hat.

Verdeckungsschwelle bzw. Maskierung

Es wird sich ein Selektionseffekt des menschlichen Ohres zu Nutze gemacht. Leise Töne werden durch gleichzeitiges Auftreten von lauten Tönen überdeckt. (simultane Verdeckung) Es gibt auch eine zeitl. Maskierung nach 10 Minuten hören von Presslufthammer hört man eine Zeit lang leise Töne nicht.

Barkhausen verwendete Maskierungseffekt um menschl. Hören in 24 kritische Bänder einzuteilen. Breite der Bänder ist nicht konstant. Verändert sich mit der mittleren Bandfrequenz. Aufgrund der Signalstärke in einem Frequenzbereich können anhand der Barkhausen-Bänder die Dauer der Maskierung errechnet und Informationen zu den verdeckten Tönen können ohne Beeinträchtigung der Hörgenusser entfernt werden.

Predictiv Coding

Ist eine Methode, bei der aus den bereits abgespielten Signalen das wahrscheinlich folgende Signal errechnet wird. Entspricht das folgende Signal nicht der Vorhersage, wird lediglich die Differenz zum vorher angenommenen Signal gespeichert. (LPAC-Codec Lossless Predictive Audio Compression)

Transform Coding
Es werden Daten in einen besser zu komprimierenden, mathematischen Raum übertragen. Fourier-Transformation von Zeit- nach Frequenzbereich.

Sub Band Coding
Beim Sub Band Coding wird das zu kodierende Audiosignal durch eine Filterbank vom Zeit- in den Frequenzbereich transformiert.

Welche Dateiformate bzw. Codecs kennen Sie?

WAV – Wave Form Audio File Format
Bei WAV handelt es sich um ein unkomprimiertes Dateiformat und die Daten werden häppchenweise in *Chunks* unterteilt und abgespeichert.

MIDI – Musical Instrument Digital Interface

Es werden Steuersignale gespeichert und Hardware erzeugt Musik. Kleine Datenmenge Qualität jedoch von Hardware abhängig. Gesang kann nicht gespeichert werden. Steuersignale können als Noten ausgegeben werden.

MP3

Eigentlicher Name MPEG-1 Layer-3 Format Motion Picture Experts Group

Wie aus dieser Abbildung ersichtlich wird, durchläuft der Audiodatenstrom einen relativ komplexen Bearbeitungsvorgang der Kodierung, der sich aber hinsichtlich der erzielbaren Kompressionsraten, also des erreichbaren Maßes an Verringerung des Speicherplatzbedarfes durchaus lohnt. Bei MP3 können durch die variablen Bitraten teilweise sogar enorme Kompressionsraten erreicht werden. So wird bei Telefonqualität mit 8 kBit/s eine Komprimierung von 96:1 erzielt. Für den Anspruch auf CDQualität wird die Audiodatei immerhin bei 128 kBit/s im Verhältnis 12:1 verkleinert.

Weitere Formate

Ogg, Rm (Realmedia-Format), WMA/ASF Windows Media Audio Codec Advanced Streaming Format, Dolby, Next/Sun Audio File Format

Kategorisierung von Bildformaten

Pixelformate

Pixelformate (auch *Rasterformate*, engl. *bitmaps* oder *pixmaps*) speichern die Bilddaten in einer Form, die der Struktur des Framebuffers ähnelt. Das heißt, das Bild wird in einzelne Bildpunkte zerteilt, und für jeden Bildpunkt wird der Farbwert gespeichert. Charakteristische Eigenschaft von Pixelformaten ist die Abhängigkeit der Detailgenauigkeit des Bildes von der eingesetzten Auflösung und Farbtiefe.
Die Dateigröße eines Bildes in einem Pixelformat ist damit direkt abhängig von der gewünschten Qualität, was Vor- und Nachteile darstellt: Einerseits braucht ein Bild von hoher Qualität viel Speicherplatz, andererseits ist die Qualität des Bildes über die Auflösung und Farbtiefe je nach Bedarf veränderbar. Bei geringerer Auflösung des Bildes und speziell bei Vergrößerungen treten vermehrt Treppeneffekte (*Alias-Effekte)* auf. Zu den bekanntesten Pixelformaten gehören *Windows Bitmap(BMP)*, *Graphics Interchange Format (GIF)*, *Joint Photographics Experts Group (JPEG)*, *Portable Network Graphics (PNG)* und *Tagged Image File Format (TIFF)*.

Vektorformate

Vektorformate speichern Bilder nicht als ein Abbild der gesamten Szene, sondern speichern vielmehr eine Beschreibung über den Aufbau der einzelnen Komponenten des Bildes. Dies können Linien, Rechtecke, Kreise usw. sein. Da der Framebuffer das anzuzeigende Bild in einem Pixelformat für den Videocontroller bereitstellt, muss das Vektorbild für die Ausgabe in ein Rasterformat umgewandelt (*gerastert, gerendert*) werden. Das Bild kann dann optimiert auf die Auflösung des Ausgabemediums gerastert werden um beispielsweise Treppeneffekte zu minimieren. Soll ein Teil des Bildes vergrößert, also *skaliert* dargestellt werden, so wird dieser Teil wieder neu mit der maximalen Auflösung des Ausgabemediums gerastert. Dadurch treten kaum Qualitätsverluste beim Vergrößern auf. Das Rastern
kostet jedoch Rechenzeit. Durch den modularen Aufbau eines Vektorbildes aus einzelnen Komponenten ist es auch relativ einfach ein Vektorbild zu verändern, indem man Beschreibungen abändert, löscht oder hinzufügt. Weiterhin brauchen Vektorformate im Vergleich zu Pixelformaten im Allgemeinen weniger Speicherplatz. Allerdings lassen sich nicht alle Inhalte effizient in Vektorformaten speichern. Je geometrischer und „künstlicher“ die Formen, desto besser lässt sich das Bild in einem Vektorformat speichern. Vektorformate sind deshalb gut geeignet, um technische Zeichnungen, Diagramme, Schriften und Symbole zu speichern. Eher ungeeignet sind Vektorformate für „natürliche“ Bilder wie Fotos mit vielen weichen Übergängen und geringer Schärfe. Bekannte Vektorformate sind *Postscript (PS)*, *Scalable Vector Graphics (SVG), Small Web Format (Flash,SWF)* und *Drawing Interchange/Exchange Format (Autocad, DXF).*

Wie wird Farbe gespeichert?

Bilder wie Graphiken und Photos werden meistens auf *Pixel-*orientierten Ausgabegeräten wie Monitoren oder Druckern dargestellt. Das bedeutet, dass die Graphiken aus einzelnen Pixeln (aus dem englischen *picture* abgekürzt *pix* und *element*), also Punkten zusammengesetzt werden. Sind die Pixel hinreichend klein und nah beieinander, ist das Auge nicht mehr in der Lage, die einzelnen Punkte voneinander zu unterscheiden. Für den Betrachter verschmelzen die einzelnen Pixel dann zu durchgehenden Formen wie z. B. Linien oder Flächen.
Je höher die Dichte der dargestellten Pixel, desto eher werden die Punkte vom Auge nicht mehr als einzelne Punkte, sondern nur noch insgesamt als Fläche erkannt. Je höher die Dichte, desto besser erscheint auch die Qualität der Darstellung. Insbesondere treten *Treppeneffekte*, die aus der Erkennung jedes einzelnen Pixels resultieren, bei höherer Pixeldichte in den Hintergrund. Die Anzahl der dargestellten Pixel pro Bild nennt man *Auflösung*; sie

errechnet sich bei rechteckigen Graphiken aus der Anzahl Pixel pro Zeile mal Anzahl Pixel pro Spalte. Die Anzahl der Farben, die jeder einzelne Pixel annehmen kann, bzw. die Anzahl der gleichzeitig in einem Bild dargestellten Farben nennt man *Farbtiefe*. Da das Auge nur begrenzt viele Farbnuancen unterscheiden kann, wird ein Bild mit einer Farbauflösung von 24 Bit meist als „natürlich“, also ohne erkennbare Farbsprünge empfunden. Die Farbauflösung von 24 Bit bezeichnet man deshalb auch als *Echtfarbdarstellung* (engl.: *True Color*). Die Darstellung des Bildes (engl.: *Frame*) auf dem Monitor wird durch den *Videocontroller* gesteuert, welcher hierzu kontinuierlich den *Bildspeicher* (engl. *Framebuffer*) ausliest. Der Framebuffer ist ein Speicherbereich, in dem für jeden Pixel, der auf dem Monitor dargestellt werden soll, der entsprechende Farbwert gespeichert wird. Die Größe des Framebuffers ist abhängig von der Auflösung und der Farbtiefe des Bildes.

Farbmodelle

Farbsinn des Menschen

Der menschliche Sehapparat ist in der Lage, Licht unterschiedlicher Wellenlänge voneinander zu unterscheiden. Jeder Wellenlänge des vom menschlichen Auge sichtbaren Bereichs zwischen 380 nm und 780 nm lässt sich einer hierbei empfundenen *Spektralfarbe*, zuordnen. Das Auge enthält drei Typen von Farbrezeptoren, die so genannten *Zapfen*. Jede der drei Typen ist nur für einen begrenzten Frequenzbereich des Lichtes empfänglich. Ein auftreffendes Photon löst abhängig von seiner Wellenlänge ein bestimmtes elektrisches Signal (*Rezeptorpotential*) aus, welches in der Netzhaut und anschließend im Gehirn weiterverarbeitet wird. Die Frequenzbereiche der drei Zapfentypen überschneiden sich, sodass durch eine gleichzeitige Betrachtung aller Rezeptorpotentiale auf eine Frequenz zurückgeschlossen werden kann. Wichtig ist hier, dass der gleiche Farbeindruck sowohl durch eine einzige Lichtfrequenz, aber auch durch Mischung mehrerer unterschiedlicher Lichtfrequenzen erzeugt werden kann, wenn sie den gleichen resultierenden charakteristischen Nervenreiz auslösen. Die drei Zapfentypen werden Rot-, Grün- und Blauzapfen genannt, obwohl diese Begriffs Blau-Zapfen machen nur ca. 9 % aller Zapfen aus. Im innersten Zentrum des Scharfsehens, der so genannten Foveola, sind gar keine Blauzapfen enthalten. Für die praktische Anwendung ist vor allem wichtig, dass mit drei Grundfarben ein Großteil des sichtbaren Farbspektrums erzeugt werden kann.

Technisch-physikalische Farbmodelle

Jedes Pixel auf einem Bildschirm wird durch drei Subpixel RGB zusammengesetzt. Da das menschliche Auge nur eine Auflösung von 1/60° hat, sieht der Mensch dann nur ein Summenpixel. Die drei Pixel R, G, B eröffnen einen 3dimensionalen Farbraum. R,G,B zusammen ergeben weiß. Man kann aber auch die Komplementärfarben Cyan, Magenta, und Gelb verwenden. Diese ergeben zusammen schwarz. Das Schwarz ist aber nicht wirklich satt, daher wird Schwarz zusätzlich verwendet. Komplementärfarben filtern ihr Komplement heraus. Cyanfarbige Pigmente auf weißem Papier verringert den Rot-Anteils des reflektierten Lichtes.

Beim YCbCr-Farbmodell werden statt drei Farbparametern zwei Farbparameter plus einen Helligkeitsparameter verwendet. Farbe wird aus den Komponenten Grundhelligkeit Y, der Abweichung Grau nach Blau bzw. Gelb Cb, und der Abweichung Grau nach Rot bzw. Türkis Cr gebildet. In einigen Bildformaten wird das YCbCr-Modell benutzt, um Schwächen des menschlichen Auges auszunutzen und Speicherplatz zu sparen, indem die Farbinformationen gezielt weniger präzise verarbeitet werden.

Wahrnehmungsorientierte Farbmodelle

Wahrnehmungsorientierte Farbmodelle versuchen der menschlichen Art des Farbempfindens entgegenzukommen, indem sie Farben nicht als Mischung anderer Grundfarben darstellen, sondern Farbinformationen und Helligkeitsinformationen getrennt betrachten. Das *HSV-Farbmodell* bestimmt eine Farbe durch den *Farbton* (*H,* engl. *hue*), der *Sättigung* (*S*, engl. *saturation*) und dem *Grauwert* (*V*, engl. *value*). Das HSV-Farbmodell wird oft eingesetzt, wenn eine bestimmte Farbe manuell reproduziert werden soll. Es wird in diesem Fall zunächst der Farbton gewählt, der durch nur einen Parameter festgelegt wird. Im zweiten und dritten Schritt werden dann Helligkeit und Sättigung dieses Farbtons gewählt.

Alphakanal

Alphakanal ist wie eine zusätzliche Achse im Farbraum. Werden mehrere Farben übereinander gelegt bestimmt der Alphakanal wie durchlässig (transparent) eine darüber liegende Farbe ist. Sinn macht ein Alphakanal, wenn verschiedene Bilder in Schichten (*Layern*) übereinander gelegt werden, dem *Alpha Blending*. Durch das transparente mischt sich die Vordergrundfarbe mit der Hintergrundfarbe. Anwendungsgebiet ist das so
genannte *Anti-Aliasing. Hierbei* versucht man die pixelig erscheinenden Kanten zu glätten, indem man einige der Randpixel transparent darstellt, und somit ihre Wahrnehmungsintensität verringert. Im Auge entsteht so der Eindruck einer gleichmäßigeren Struktur, die einzelnen Pixel werden weniger stark wahrgenommen. Dass die gängigen Bildformate eine rechteckige Grundform vorschreiben, macht es schwierig, beliebige Objekte mit beliebigen Hintergründen zu vereinen. Auch hier hilft der Alphakanal, indem man sämtliche Bereiche, die nicht zum eigentlichen Objekt gehören, zu 100 % durchsichtig markiert. Anstelle der entsprechenden Bereiche ist dort nur noch der Hintergrund sichtbar.

Echtfarbdarstellung und Palette

Wie bereits erwähnt, wird eine Farbtiefe mit 24 Bit Farbtiefe, also ca. 16,7 Millionen Farben als ausreichend angesehen, um einen natürlichen Farbeindruck zu gewinnen. Allerdings unterstützt nicht jedes Ausgabegerät eine so hohe Farbanzahl. Häufig ist es so, dass aufgrund von Speicherplatzbeschränkungen nur eine Auswahl von Farben *gleichzeitig* darstellbar ist. Diese Auswahl nennt sich Palette. Die Palette ist eine Datenstruktur, die Farbwerte (z. B. RGB Tripel) auf einen Index abbildet. Die eigentlichen Pixeldaten speichern dann Indexwerte statt der Farbwerte. Viele Bildformate unterstützen ebenfalls Farbpaletten. Der Vorteil der Farbpalette ist der geringere Speicherplatz pro Pixel, der für nur von der Größe der Farbpalette abhängt.

Nennen Sie Kompressionsverfahren!

Man unterscheidet verlustfreie und verlustbehaftet Kompression. Verlustfrei bedeute dass die Ursprungsdaten zu 100% wieder hergestellt werden können. Verlustfrei kann ohne Kenntnis über Inhalt und Bedeutung der Daten durchgeführt werden.

Verlustfreie sind Lauflängenkodierung, die Lempel-Ziv Familie, die Huffman-Kodierung und die Diskrete Kosinus-Transformation.

Läuflängenkodierung

Lauflängenkodierung (engl. *Run Length Encoding, RLE*) ist ein sehr einfacher Kompressionsalgorithmus. Er fasst aufeinander folgende gleiche Datenwerte zusammen und ersetzt sie durch die Anzahl des Auftretens (Lauflänge) und den Wert selbst. Geeignet ist die Lauflän-

genkodierung deshalb vor allem für Bilder mit wenigen Farben und vielen Flächen. Verwendet wird die Lauflängenkodierung bei Windows Bitmap (RLE/BMP).

Huffman-Kodierung

Die Huffman-Kodierung ist eine *Entropiekodierung.* Sie kodiert Zeichen mit einer variablen Anzahl von Bits. Hierzu wird eine *präfixfreie* Kodierung verwendet, bei der also kein Kodewort Präfix eines anderen Kodewortes sein darf. Bei der Huffman-Kodierung erhalten die häufiger auftretenden Zeichen die kürzeren Kodierungen und die selten auftretenden Zeichen die längeren Kodierungen. Huffman-Kodierer werden häufig mit anderen Kompressionsverfahren kombiniert, wie z. B. mit Verfahren aus der LZ-Familie.

Lempel-Ziv Familie

Die Lempel-Ziv Familie, zu der *LZ77*, *LZ78*, und *LZW* (Lempel-Ziv-Welch) gehören, sind *Stringersatzverfahren.* Sie benutzen ein „Wörterbuch“ um wiederkehrende ganze Wörter in der Datenmenge zu indizieren, um dann nur noch einen Index anstelle des Wortes zu speichern. LZW wird z. B. bei GIF (optional: TIFF und JPEG) verwendet. Für LZW (nicht aber LZ77) machte die Firma Unisys zusammen mit GIF-Erfinder Compuserve Patentansprüche geltend und verlangte bis zum Ablauf des Patents im Jahr 2003 Lizenzgebühren von Herstellern von Softwareprodukten, die GIF-Dateien schreiben konnten. Das unpatentierte LZ77 wird von PNG benutzt.

Verlustbehaftete Kompressionsmethoden

Bei der verlustbehafteten Kompression (*Irrelevanzreduktion*) wird ein gewisser Verlust von Information in Kauf genommen um eine noch höhere Kompression als bei der verlustfreien Kompression zu erreichen. Die ursprüngliche Information kann nicht wiederhergestellt werden. Eine verlustbehaftete Kompression ist immer auf den Inhalt und den Typ der Daten abzustimmen. Verlustbehaftete Kompression wird vor allem bei Bild- und Tondateien eingesetzt. Hier macht man es sich zu Nutze, dass die menschlichen Sinne über eine begrenzte Leistungsfähigkeit verfügen und bestimmte Informationen gar nicht verarbeitet werden können. Man kann beispielsweise die begrenzte Fähigkeit zur Unterscheidung zwischen kleinen und nah beieinander liegenden Punkten nutzen, um auf dem Monitor oder Papier die Illusion von zusammenhängenden Figuren zu erzeugen. Begrenzt ist auch die Fähigkeit, ähnliche Farbtöne voneinander zu unterscheiden.

Bildformate

BMP - > Windows Bitmap

BMP wird zwar von sehr vielen Anwendungen unterstützt, aufgrund der Größe der unkomprimierten Bilddateien jedoch nur selten eingesetzt.

GIF -> Graphics Interchange Format

GIF87a wurde 1987 durch den Online-Anbieter Compuserve entwickelt, um Benutzern durch kleinere Dateigrößen einen schnelleren Download von Bildern über den Onlinedienst zu ermöglichen. Der *Interlace-Modus* ermöglicht es, schon während des Ladevorgangs einen groben Eindruck über den Bildinhalt zu bekommen. GIF-Interlace baut das Bild hierfür zeilenweise auf. Bei jedem Iterationsschritt wird hierbei die horizontale Auflösung verdoppelt. GIF nutzt das LZW-Verfahren zur Kompression. Deshalb, und wegen der Beschränkung auf

256 Farben, ist GIF nicht für fotorealistische Bilder geeignet.

PNG -> Portable Network Graphics (*PNG is Not Gif)*

Ziel von PNG ist es, einen Ersatz für das einfachere GIF sowie in einem gewissen Rahmen, für das viel komplexere TIFF zu schaffen.

JPEG -> Joint Photographics Experts Group

https://www.univie.ac.at/video/grundlagen/dct.htm

Diskrete Cosinustranformation (DCT): Ein Bild kann man als Folge von Zahlenwerten (=Farbinformationen) sehen. Trägt man diese Zahlenwerte der Reihe nach in einem Koordinatensystem ein, so erhält man eine Funktion. Diese Zahlenwerte können durch eine leicht darstellbare Funktion angenähert werden. Die DCT ist eine Variation der Fouriertransformation, die beliebige Signale (jede Folge von Zahlenwerten) durch Überlagerungen von Cosinuswellen mit verschiedener Frequenz und Amplitude darstellt. Die Pixelwerte werden als Frequenz und Amplitudenverteilung dargestellt. Dabei werden die unterschiedlichen Frequenzen des Signals sichtbar. Unser Auge ist für niedrigere Frequenzen weitaus empfindlicher als für höherfrequente. Indem man höherfrequente Anteile vernachlässigt, kann die Datenmenge reduziert werden. Es geht zwar Information verloren, das Auge kann die Unterschiede aber kaum erkennen. Anstatt der Zahlenwerte (Pixel), werden in der Folge nur die Parameter der Funktionenreihe übertragen. Ein Bild wird in 8x8-Pixel-Blöcke zerlegt, die jeweils einer DCT unterzogen werden. Für jeden Block wird eine 8x8-Koeffizientenmatrix berechnet.

• Quantisierung: Die nun ermittelten Koeffizienten werden so quantisiert, dass die Koeffizienten hochfrequenter Bildteile Null ergeben.

• Entropie-Codierung: Die ermittelten Werte (Funktionskoeffizienten) werden in eine geeignete Reihenfolge (sodass eine Zahlenreihe mit vielen Nullen am Schluss entsteht) gebracht und dann RLE-codiert (Lauflängencodierung). In einem nächsten Schritt wird dieses Zwischenergebnis der Huffman- oder arithmetischen Codierung unterzogen.

Das umgekehrte Verfahren, das Errechnen von Bildpunkten aus den DCT-Koeffizienten, heißt Inverse Diskrete Cosinustransformation (IDCT).

http://jendryschik.de/weblog/2002/06/03/diskrete-cosinus-transformation-dct

http://www.inf.fh-flensburg.de/lang/algorithmen/fft/dct.htm
DXF -> Drawing Exchange Format

Wie werden Videos komprimiert?

Ziel von Videokompression ist es, jede nicht im Originalsignal benötigte Information zu beseitigen und damit Bandbreitenbedarf zu senken, ohne jedoch bei qualitativer Beurteilung wesentliche Unterschiede erkennen zu lassen. Basis für die Reduktion von Videosignalen ist das Vorhandensein von Ähnlichkeiten im Bild. Bei zwei zeitlich aufeinander folgenden Bildern wird deutlich, dass sich der Bildinhalt von Bild zu Bild nur wenig ändert. Oft können wir feststellen, dass diese Änderung nur von der Bewegung einzelner Objekte herrührt.
Zwei Datenreduktionsprinzipien sind Entropiekodierung und Quellenkodierung, wobei wir in der Praxis zumeist kombinierte Anwendungen finden. Die Quellenkodierung lässt sich noch in prädiktive und frequenzorientierte Kompression unterteilen.

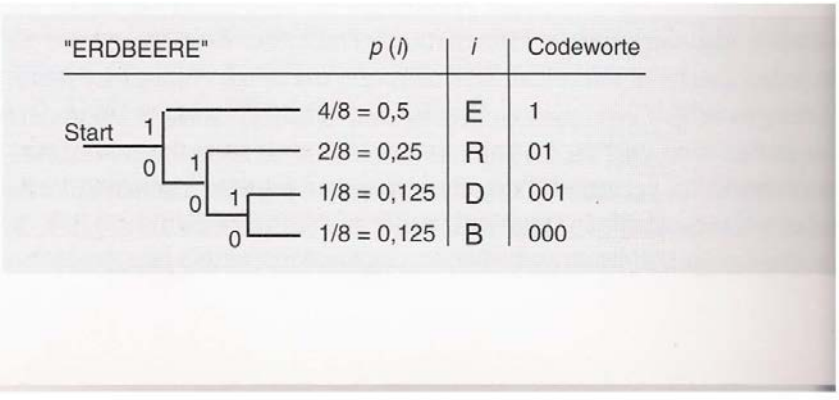
Entropiekodierung beseitigt Redundanz. Sie ist verlustfrei. Zwei Vertreter sind VLC und RLC.

Entropiekodierung: Variable Length Coding (VLC)

Es wird sich der Huffman Algorithmus zu Nutze gemacht. Häufig vorkommende Zeichen werden mit kurzer Bitfolge kodiert. Weniger häufig vorkommende Zeichen werden mit einer längeren Bitfolge kodiert. Es werden den Zeichen Einzelwahrscheinlichkeiten zugewiesen die werden beginnend vom den unwahrscheinlichsten paarweise zusammengefasst. An jedem Kreuzungspunkt wird mit 1 bzw. 0 verzweigt.

Die Einzelwahrscheinlichkeit der seltensten Zeichen D und B werden zu einer gemeinsamen Wahrscheinlichkeit addiert. Dieser Prozess wird solange wiederholt bis nur noch zwei Häufigkeiten übrig bleiben, denen die Codewörter 1 und 0 zugewiesen werden. Dann werden alle Stufen zurückverfolgt, wobei jeder Knotenpunkt die Zweigstelle zum nächst längeren Codewort bildet. Auf Empfängerseite muss der Prozess bekannt sein. Dies ist in Form einer festgelegten Huffman-Tabelle möglich, die dem Decoder bekannt ist und mit dem Datenstrom mitgeliefert wird.

Abb.2. Beispiel für eine variable Längecodierung nach Huffmann



Entropiecodierung: RLC (Run Length Coding)

Weitere Verbesserung der Datenreduktion kann durch RLC erzielt werden. Viele Daten bestehen aus einer Folge identischer Bytes. Je größer deren Anzahl ist, desto besser ist die Datenreduktionsrate. Eigentlicher Algorithmus zählt wiederholende Bytes und stellt eine Markierung, die nicht als Datenbestand vorkommt, vor der ersten wiederholenden Byte - Folge. Hiermit kann der Beginn einer solchen Lauflängenkodierung markiert werden. Auf einem Beispiel kann deutlich erklärt werden:

Unkomprimierte Daten : **ABCCCCCCCCCCCCDEFGGG**
Lauflängenkodierung(RLC) : **ABC!8DEFGGG**
In diesem Beispiel wurde die Vereinbarung getroffen, dass erst nach vier wiederholenden Bytes weitere identische Bytes gezählt werden. Dazu kommt eine Markierung (Aufrufzeichen) und die Zahl der wiederholenden Bytes. In diesen Fall wurden 12 Zeichen auf 3 komprimiert. Diese Kodierung kann auf verschiedene Art gelöst werden und Absprache muss dann sowohl bei der Kodierung als auch bei der Dekodierung bekannt sein. Beide Verfahren können unabhängig von anderen eingesetzten Verfahren angewendet werden.

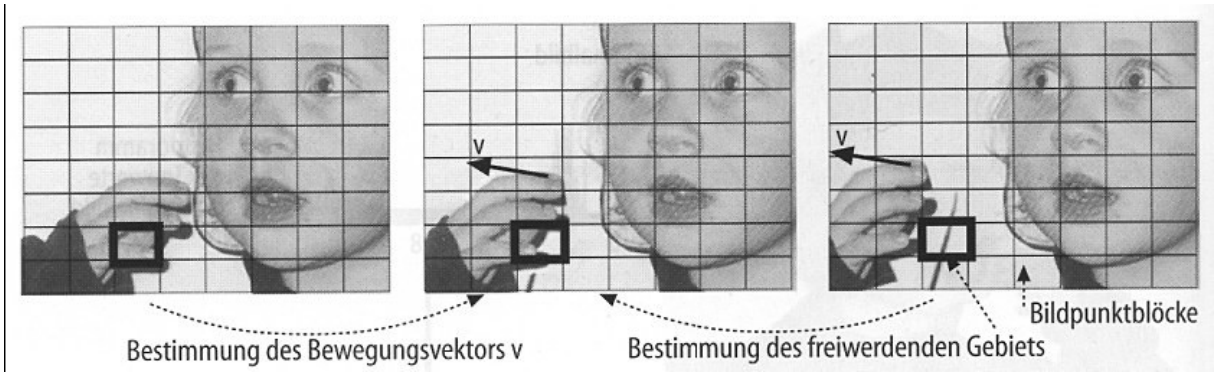
Quellenkodierung

Quellenkodierung benutzt Irrelevanzreduktion, die eine Beschneidung der kodierten Information ist. Es werden Signalteile entfernt, die bei späterer Dekodierung nicht wieder hergestellt werden können. Welche Informationen als irrelevant erachtet werden, hängt von der jeweiligen Anwendung ab.

Quellenkodierung: DPCM (Differenzielle Puls Code Modulation)

„Prädiktion“ oder „relative Kodierung“ bezeichnet ein Datenreduktionsverfahren, welches auf dem Prinzip der „Differenzkodierung“ beruht. Dabei werden nur ein Bild und die Differenz zum nächsten oder vorherigen (je nach Implementierung) Bild gespeichert, anstatt beide jeweils komplett einzeln zu speichern [40].
Die *Differential Pulse Code Modulation (DPCM)* nutzt großflächige Regelmäßigkeiten eines Bildes aus, indem nur die Differenzen zwischen aufeinander folgenden Bildern oder zwischen Nachbarpixeln eines Bildes gespeichert werden. Bei großen Flächen sind die Differenzen benachbarter Pixel gering, bei Kanten groß. Die Differenzbildung lässt sich als Vorhersage (Prädiktion) beschreiben, die aus den vorherigen Bildern oder Pixels auf das nächste getroffen wird [43]. Mittels einer Vorhersagefunktion wird eine Prädiktion (Schätzwerte der Bilddaten) des folgenden Bildes (oder benachbarten Pixels) errechnet, gespeichert wird aber die Differenz aus dem realen Wert des folgenden Bildes (oder benachbarten Pixels) und der Prädiktion. Haben zwei Pixel beispielsweise die Grauwerte 194 und 209, die mit 8 Bit kodiert werden können, so ergibt die Differenz einen Wert von 15, welcher mit nur 4 Bit dargestellt werden kann. Diese Differenz wird auch als *Prädiktionsfehler* bezeichnet. Die mittels der Vorhersagefunktion geschätzten Werte liegen näher an den Werten des Nachbarpixels oder Folgebildes als die Originalwerte des vorherigen Pixels oder Bildes. Daher können kleinere Differenzen erzielt werden, was eine stärkere Datenreduktion bedeutet.
Die DPCM kann also auf Einzelbilder (Intraframe-DPCM) und auf zeitlich benachbarte Bilder (Interframe-DPCM) angewendet werden. Letzteres steigert die Effizienz der DPCM, da in der Regel zwei aufeinander folgende Bilder größere Ähnlichkeiten aufweisen als benachbarte Pixel [39]. Aufeinander folgende Bilder unterscheiden sich hauptsächlich dadurch, dass sich Bildbereiche oft nur bewegen anstatt sich zu verändern. Da sich Bewegungen selten von Bild zu Bild ändern, lässt sich die Position eines Objekts, dessen Bewegung innerhalb eines bestimmten Zeitintervalls die Richtung nicht geändert hat, im darauf folgenden Bild mit sehr großer Wahrscheinlichkeit abschätzen. Mithilfe der Bewegungsvorhersage aus dem Vergleich von aktuellem und vorherigem Bild kann eine *Bewegungskompensation* auf das aktuelle Bild angewendet werden. Es wird gemäß der Bewegungsvorhersage verändert und ist dadurch seinem Nachfolger wesentlich ähnlicher, was die Differenzbildung begünstigt. Zum Auffinden von Ähnlichkeiten im Bild wird das *Blockmatching-Verfahren* benutzt, welches auf Bildpunktblöcke angewendet wird um einen ökonomischen Rechenaufwand zu gewährleisten. So wird im Nachbarbild nach dem ähnlichsten Block gesucht und der relative Verschiebungsvektor zum Ausgangsblock gespeichert (Abbildung 4.3). Diese Art der Prädiktion birgt allerdings Fehlerquellen; so kann nicht vorhergesagt werden, ob ein Objekt vorher nicht

sichtbare Teile des Hintergrunds nach der Vollendung seiner Bewegung freilegt. Zur Lösung dieses Problems kann eine bidirektionale Prädiktion benutzt werden, die sich auf das vorherige und das nachfolgende Bild bezieht [39].



Die DPCM kann also auf Einzelbilder (Intraframe-DPCM) und auf zeitlich benachbarte Bilder (Interframe-DPCM) angewendet werden. Letzteres steigert die Effizienz der DPCM, da in der Regel zwei aufeinander folgende Bilder größere Ähnlichkeiten aufweisen als benachbarte Pixel [39].

Quellencodierung: DCT (Diskrete Cosinus Transformation)

Bei der Transformationskodierung werden Daten in einen anderen mathematischen Raum transformiert, wodurch sie effektiver komprimiert werden können. Eine der effektivsten Transformationen zur Datenreduktion ist die *Diskrete Kosinus-Transformation* (DCT), die auch in JPEG und MPEG Anwendung findet [40].
Mit der DCT werden die Pixelwerte eines Bildblocks aus dem Ortsbereich in den Ortsfrequenzbereich transformiert (Abbildung 4.4 und Abbildung 4.5). Grobe Bildstrukturen und langsame Helligkeitsübergänge (Flächen) werden durch tiefe Ortsfrequenzen repräsentiert, kleine Bildstrukturen und schnelle Helligkeitsübergänge (scharfe Kanten) durch hohe Ortfrequenzen. Der Fokus liegt dabei auf den niederfrequenten Komponenten, da diese in natürlichen Bildern häufiger vorkommen als feine.
Das Weglassen von hochfrequenten Komponenten produziert irrelevante, oft nicht oder kaum sichtbare Fehler. Zur DCT wird das Bild in quadratische 8x8- Pixelblöcke aufgeteilt und die Transformation auf jeden dieser Blöcke angewandt. Die Transformation eines Pixelblocks produziert einen gleich großen Block mit DCT-Koeffizienten. Die DCT arbeitet prinzipiell verlustfrei und ist vollständig reversibel, allerdings werden die Koeffizienten in der folgenden Quantisierungsstufe gerundet. Die DCT ist also 64-mal pro Pixelblock anzuwenden. In der linken oberen Ecke der DCT-Matrix befindet sich die *DC-Komponente* (Gleichspannungsanteil) für den mittleren Grauwert des Blocks und bestimmt den Grundton des Pixelblocks. Rechts und darunter befinden sich die Koeffizienten für die niederfrequenten AC-Komponenten (Wechselspannungsanteil). Die unbedeutenderen, hochfrequenten AC-Komponenten stehen im rechten, unteren Bereich. Dabei steht oben rechts der Koeffizient für die höchste Frequenz in waagerechter Richtung, links unten der für die höchste Frequenz in senkrechter Richtung. Die hochfrequenten AC-Komponenten weisen normalerweise niedrigere Werte, meist nahe Null, auf, wodurch längere Folgen ähnlicher Bytes entstehen [40]. Darüber hinaus sind die Unterschiede der Zahlenwerte in der DCT-Matrix weitaus größer als in dem Originalblock. Es zeigt sich, dass in der DCT-Matrix ein hoher DCT-Koeffizient steht

und die meisten restlichen Werte nahe Null liegen. Dies bietet gute Voraussetzungen für RLC und VLC [43].

Der bisher beschriebene Vorgang (FDCT – Forward-DCT) muss bei der Dekodierung umgekehrt werden (IDCT – Inverse-DCT) [40].

Quantisierung und inverse Transformation (IDCT)

Mit der *Quantisierung* aller durch die DCT gewonnenen Werte soll eine weitere, aber verlustbehaftete Datenreduktion vorgenommen werden.
Dies geschieht durch das Herabsetzen der Genauigkeit der Daten mittels Division und Rundung, meist auf ganze Zahlen. Eine *Quantisierungstabelle* von der Größe der DCT-Matrix (8x8) stellt pro DCTKoeffizient einen 8 bit-kodierten ganzzahligen Eintrag als Divisor bereit [40]. Damit kann jeder DCTKoeffizient separat und unterschiedlich eingestellt und bewertet bzw. gewichtet werden (*Gewichtungsmatrix*, Abb. 6). Ein kleineres Gewicht bedeutet einen größeren Divisor [43]. Der Kompressionsgrad kann so auf Kosten der Bildqualität angepasst werden. Die Quantisierung erfolgt gemäß der Formel:
Quantisierter DCT-Koeffizient = round (DCT-Koeffizient/Divisor)

Was ist Musikretrieval?

Das ist die Suche nach Musikinhalten welche sich aber nicht auf die Suche nach manuell erstellten Metadaten beschränkt sondern auf die Inhalte selbst sucht.

„*Musik- Retrieval stellt ein interdisziplinäres Forschungsgebiet dar, das bereits seit Ende 1960 existiert und sich mit der zunehmenden Verbreitung des Internet und entsprechender digitaler Formate (bspw. MP3) zunehmend etablieren konnte. Die zu lösenden Kernprobleme umfassen Technologien, die dem Menschen einen effizienten Zugriff auf umfangreiche Musik-kollektionen ermöglichen.“*

Es werden an Musik-Retrieval-Systeme Anforderungen gestellt wie die Merkmals-Extraktion aus Musik- und zugehörigen Audio-Signalen und dort z. B. das Erkennen von Akkorden, die automatische Segmentierung der Audioströme, die Klassifizierung der Inhalten nach Musikarten bzw. Musikrichtungen sowie die Indexierung derselben auf der Grundlage einer solchen Klassifikation und anderer Methode. Darüber hinaus stehen im Fokus des Interesses die Entwicklung von intuitiven Benutzungsschnittstellen für die Musiksuche, die Nutzung psycho-akustischer Phänomene (ähnlich wie bei der Audiokodierung, siehe Kurs 1875 „Multimediainformationssysteme I“, Kurseinheit 1), die Anordnung der Audiodaten nach Ähnlichkeit und die geeignete Darstellung von Suchergebnissen.

Hinsichtlich seiner Komponenten besteht ein Musik-Retrieval-System aus mehreren Datenbanken zur Speicherung und Verwaltung der Musikwerke sowie den entsprechenden Meta-Daten, einem Server für die Bearbeitung der Benutzeranfragen und die Verwaltung der Datenbanken und eines geeigneten Werkzeuges zur Unterstützung der inhaltsbasierten Suche.

Der Inhalt einer Musikdatei lässt sich im inhaltsbasierten Musik-Retrieval durch verschiedene Arten von Merkmalen charakterisieren. Welche im Rahmen eines Suchvorgangs verwendet werden können. Es kommen dabei verschiedene Dimensionen in Betracht. Die Tonhöhe, die Dauer von Tönen und das Tempo, mit der aufeinanderfolgende Töne erklingen, die musikalische Harmonie zwischen tönen, die Klangfarbe. Diese werden in Musikdatenformaten in der Regel mithilfe des Musical-Instrument-Digital-Interface-Standards (MIDI) kodiert.

Für das Musik-Retrieval gibt es verschiedene Möglichkeiten, wie eine Anfrage durchgeführt werden kann.

Zum einen die direkte Eingabe von Musik-Merkmalen als Grundlage für eine merkmals-basierte Suche. Tonhöhen und Tempi lassen sich mit einer geeigneten Anfragesprache eingeben.
Daneben gibt es die Beispielssuche (query by example). Es wird dabei eine Sequenz eines Musikstückes vorgespielt. (Beispiel Shazam) Um Musikstücke finden zu können werden spezifische Eigenschaften jedes Musikstrückes gespeichert, die als „fingerprints“ oder „landmarks“ auf der Grundlage von verschiedenen Merkmalen wie z B. der Häufigkeitsverteilung der verwendeten Tonhöhen analysiert und dann als Such- bzw. Beschreibungsmuster in Form von Histogrammen hinsichtlich der Häufigkeiten des Auftretens der Merkmale zusammengesetzt werden.

Eine weitere Möglichkeit besteht darin, Anfragen durch Pfeifen oder Vorsummen (query-by-humming) zu formulieren. Es müssen dabei Fehler des Benutzers ausgeglichen werden. Die Schwierigkeit besteht bei dieser Art im Erkennen einzelner Töne und deren Sequenzen.

Aus dem wie erwähnt eingeben Musikfragment wird ein zugehöriges Anfrage-Objekt erstellt. Dieses Anfrage-Objekt wird dann auf geeignete Art und Weise mit der Information aus Musikdatenbanken verglichen. Hier finden Vergleichsalgorithmen ihre Anwendung. Die Rückgabe erfolgt in einer Listen mit ähnlichen Musikwerken.

In einigen Musik-Retrieval-Systemen wird eine Ergebnisliste zurückgeliefert, in der die Benutzer einen Titel auswählen können und direkt an die Datenbank schicken können, damit der Titel abgespielt wird.

In anderen Systemen, gibt es die Zusammenstellung einer Spielliste als Ergebnisliste.
Welche Ergebnisse dabei für die Präsentation in Betracht kommen, hängt dabei von den Eigenschaften der verwendeten Matching-Algorithmen ab. In [3] wird als Vergleichsmethode zunächst die Klasse der *zeichenkettenbasierten Methoden*, die sich insbesondere für monophonische Musik gut eignen, genannt. Die Musik ist bei diesen Verfahren durch Zeichenketten, die in der Regel zur Tonhöhensequenz korrelieren, repräsentiert. Die Berechnung von Distanzen erfolgt durch die Berechnung von *Zeichenkettendistanzen* (engl. *String Distances*). Es wird in der Regel nach der größten gemeinsamen Teilsequenz gesucht oder nach der Stelle, an der eine Zeichensequenz in einer anderen vorhanden ist. Darüber hinaus gibt es die *mengenbasierten Methoden.* Hier wird Musik als Menge von Ereignissen bzw. Merkmalen mit bestimmten Eigenschaften (Tempo, Rhythmus, Tonhöhe) verstanden. Abschließend sind die *probabilistischen Verfahren* zu nennen, bei denen Wahrscheinlichkeiten für das Auftreten von musikalischen Ereignissen in Musikstücken als Grundlage für das matching verwendet werden.
Diese unterschiedlichen Klassen von Matching-Methoden und deren Indexstrukturen sind also für die Effektivität und die Effizienz der Suche verantwortlich. Am häufigsten werden *R-Bäume* und *invertierte Listen* für die Erstellung von besonders effizienten Indexstrukturen verwendet [3].

MPEG7

Wozu?
Aus einem Medienobjekt können bestimmte beschreibende Daten entweder extrahiert oder durch manuelles Annotieren erzeugt werden. MPEG-7 stellt einen Standard für die Repräsentation dieser Meta-Daten zur Verfügung. Diese Metadaten können so zum Beispiel durch Suchmaschinen indexiert und von Anwendungen gefunden, adressiert oder mit anderen Objekten verglichen werden.

Die Erzeugung von inhaltsbeschreibenden Metadaten erfordert in der Regel semi-automatische Extraktionsmethoden, die einerseits alles, was automatisiert aus einem Medienobjekt herausgezogen werden kann (wie z. B. die Bildgröße und Auflösung, aber auch Farbinformationen über Videosegmente etc.) erfassen und andererseits den Nutzern die Chance geben, weitere Daten manuell zu erschließen (wie die Namen der auf einem Bild dargestellten Personen) und der Beschreibung der automatisch extrahierten Merkmale hinzuzufügen.

Aus Multimedia-Objekten können bestimmte Medadaten gewonnen werden. Dies kann wie bereits erwähnt automatisiert und/oder manuell erfolgen. Solche Metadaten können dann, gesetzt den Fall, dass diese in einem standardisierten Format vorliegen, als Datenbasis für eine Multimedia-Datenbank dienen, in welcher Mediendaten verwaltet und entsprechend abgefragt und gefunden werden können.

Um eben eine standardisierte Darstellung dieser Metadaten geht es in MPEG-7. Diese Daten werden als XML-Datei repräsentiert.

Was ist es?

Einzelne Bestandteile erläutern!

Der MPEG-7-Standard besteht aus den acht Teilen Systems, Data Definition Language (DDL), Visual Description, Audio Description, Multimedia Description Schemes (MDS), Reference Software, Conformance Testing, Extraction and Use of Descriptions, die im Folgenden kurz erläutert werden sollen. Der System-Teil von MPEG-7 definiert die binäre Kodierung und Auslieferung von MPEG-7 Dateien. Teile eines Multimedia-Objektes können in diesem Teil zunächst lose mithilfe eines Deskriptors (engl. descriptor) beschrieben werden. Diese Teilbeschreibungen werden dann durch Schemata (engl. schemes) innerhalb eines Descriptor Schemas in Relation zueinander gesetzt. Die Data Description Language im zweiten Teil des Standards beschreibt, wie Deskriptoren und Deskriptoren-Schemata erzeugt werden können und gibt die Syntax für deren Repräsentation vor. Im dritten und vierten Teil des Standards wird die Beschreibung von reinen Video- bzw. Audioobjekten standardisiert. Der fünfte Teil standardisiert die semantische Beschreibung weiterer Meta-Daten. Die Teile 6 und 7 kümmern sich um die Implementierung des Standards. Im sechsten Teil wird zunächst eine Referenzsoftware, die die anderen Teile realisiert, bereitgestellt, während im siebten Teil Testverfahren zur Zertifizierung, dass eine Implementierung auch standardkonform ist, bereitgestellt werden. Der achte Teil des Standards beinhaltet eine Anleitung für den Umgang mit inhaltsbasierten Deskriptoren. Die bereits erwähnte Software CaliphEmir stellt z. B. eine solche Implementierung zur Verfügung. Es handelt sich dabei um eine Software, die automatisiert Metadaten wie Farbinformationen aus Bildern herauszieht und zugleich eine graphische Benutzungsschnittstelle anbietet, in der weitere Daten eingetragen werden können. Resultat ist dann eine MPEG-7-Datei (Dateiendung lautet *.mp7.xml). Zudem erlaubt es die Software auch, nach bestimmten Daten innerhalb solcher MPEG-7-Dateien zu suchen. Weitere Anwendungen, die MPEG-7 implementieren sind z. B. das IBM VideoAnnEx Annotation Tool (http://www.research.ibm.com/VideoAnnEx/) sowie z. B. der von der TU-Berlin bereitgestellte MPEG-7-Audio-Analyzer (http://mpeg7lld.nue.tu-berlin.de/)

OAIS?

Was ist das?
OAIS definiert also ein konzeptuelles Rahmenmodell für ein System zur Archivierung elektronischer Dokumente. Dieses definiert zunächst die grundlegenden Begrifflichkeiten sowie die Hauptaufgaben eines solchen Systems und die Richtlinien für dessen Einsatz und Handhabung. Dadurch soll ein einheitliches Grundverständnis von der Thematik für den globalen Informations-und Erfahrungsaus-tausch geschaffen werden. Der Schwerpunkt liegt dabei auf der Definition eines allgemeinen, konzeptuellen, theoretischen bzw. semantischen Modells für die Funktionalität eines Archivs. So besteht die Architektur des OAIS-Modells aus drei zugehörigen Modellen, dem *OAIS-Prozessmodell*, dem *OAIS-Informationsmodell* und dem *OAIS-Datenmodell*.

Beispiel?
Kopa wurde auf der Basis von DIAS-Core (IBM) entwickelt. Universitätsbibliothek Göttingen und die Deutsche Nationalbibliothek erweiterten DIAS um Funktionalitäten zur Erfassung von Meta- und Archivdaten. Ein Access-Tool ermöglicht den strukturierten Zugriff auf den Datenbestand.

Zur Einarbeitung der Daten kommt ein Ingest-Tool zum Einsatz. Für SIP und DIP bestehen eigene Definitionen.

BABS vom Leibniz-Rechenzentrum und der Bayrischen Staatsbibliothek.

Bei *Fedora* (The Flexible Extensible Digital Object and Repository Architecture) handelt es sich um ein open-source Projekt der Cornell University, Department of Information Science und der Library der University of Virginia.

Prozesse

Die allgemeinen Dienste (Common Services) umfassen die betriebssystemspezifischen Dienste, die Kommunikation von mehreren Prozessoren untereinander, die Netzwerkdienste, den Dienstnamenservice, das Management des temporären Speichers, die Dienste zur Fehlerbehandlung und die Dienste zur Gewährleistung von Sicherheit.

Datenübernahme (Ingest) nimmt die SIPs vom Erzeuger entgegen. Zu seinen Aufgaben gehört die Speicherung dieser Information im System und die Aufbereitung dieser Daten für weitere Verarbeitung in das interne Format. Weiterhin werden die Daten vom Ingest Process auf Vollständigkeit und Korrektheit validiert. Um einen authentifizierten Zugang zum System zu gewährleisten, muss sich ein Erzeuger gegenüber dem System authentifizieren, wodurch ein Kontrollmechanismus für den Zugriff geschaffen wird. Die Durchführung der Authentifizierung ist eine weitere Aufgabe vom Ingest Process. Der geprüfte SIP wird in AIP anschließend umgewandelt und die zugehörige Descriptive Information wird aus dem SIP generiert. Die generierte Descreptive Information wird dann im Data Management abgelegt.

Archivspeicher (Storage) ist für die Verwaltung, Speicherung und Zur-Verfügungstellung der Daten im Originalzustand zuständig. Der Archival storage ist für die Belieferung des Access mit den angeforderten Daten zuständig.

Datenverwaltung (Management) pflegt die Descriptive Information, die vom Ingest Process aus den SIP erzeugt wurden. Zur Speicherung der Daten wird (im Gegensatz zu Archival storage) immer eine Datenbank eingesetzt. Die Verwaltung dieser Datenbank ist die Aufgabe des Data Managements. Weiterhin müssen die gelieferten Daten für die Speicherung aufbereitet werden. Die Abfragen der Informationen können nur über das Data Management erfolgen, da nur dieser einen Zugriff auf die Descriptive Information zu den gesamten Daten hat.

Zugriff (Access) ist derjenige Prozess, der das Bindeglied zwischen dem System und den Verbrauchern ist. Er stellt die Schnittstelle zu den Benutzern dar, nimmt die Suchanfragen entgegen und bereitet die erhaltenen Ergebnisse vom Archival Storage entsprechend der Schnittstellen-beschreibung auf. Weiterhin überwacht dieser Prozess die korrekte Auslieferung erzeugter Information an die Verbraucher.

Die Aufgabe der Erhaltunsplanung (Preservation Planning) besteht in der langfristig orientierten Archivierung der Daten, damit zukünftig ein Zugriff auf diese Daten immer noch möglich ist. Hierfür müssen die aktuellen Veränderungen und Trends der Hardware und Software auf dem Markt verfolgt werden, um rechtzeitig notwendige Handlungsmaßnamen herzuleiten und diese umzusetzen. Eine davon kann z. B. die *Migration* der Daten sein.

Was ist OAI-PMH (Protocol for Metadata Harvesting)?

Protokoll zum Zugriff auf OAIS Daten. Der Zweck von OAI-PMH besteht darin, Dokumente für unterschiedliche Benutzergruppen einheitlich zu veröffentlichen. Der größte Einsatz von OAI-PMH findet an den Universitäten in der Publikation wissenschaftlicher Dokumente statt.

Was ist METS?

Der *Metadata Encoding and Transmission Standard* (METS) [23] gibt vor, die Metadaten in einem auf XML basierenden Format zu speichern. METS wurde entwickelt, um sowohl digitale Objekte innerhalb einer Bibliothek zu verwalten als auch den Austausch dieser Objekte mit anderen Bibliotheken oder Benutzern zu gewährleisten. Als generisches Containerformat kann METS daher grundsätzlich für alle Datentypen und –formate verwendet werden. Die sieben Hauptabschnitte einer METS-XML-Datei beinhalten u. a. beschreibende (also inhaltliche), administrative sowie strukturelle Metadaten. Ebenfalls sind technische und Rechtemetadaten hier als Teil der administrativen Meta-daten berücksichtigt. Somit werden die oben genannten Metadatentypen grundsätzlich von METS abgedeckt.

Was ist Dublin Core?

Per *Dublin Core Metadata Element Set* [24] setzt sich allmählich auch im Internet ein Standard für die Beschreibung von Ressourcen (z.B. HTML-Seiten) mithilfe von Metadaten durch. Das Dublin Core Metadata Element Set bestehtdabei aus mindestens 15 Elementen, also dem *Simple Dublin Core*, welcher von der *Dublin Core Metadata Initiative* (*DCMI*) empfohlen werden. Diese Elemente lassen sich ebenfalls wieder anhand der oben genannten Typen wie folgt kategorisieren: Inhaltsmetadaten (z. B. Titel, Thema, Beschreibung), technische Metadaten (Typ, Format), strukturelle Metadaten (Quelle, Bezug), administrative Metadaten (Datum) sowie Rechtemetadaten, wobei primär Inhalts-metadaten berücksichtigt werden und die anderen Typen nur teilweise abdeckt werden. Dublin Core Metadaten können sowohl in das Dokument eingebettet werden (wie bei HTML-Seiten oder anderen XML-basierten Dokumenten) oder in externe Datenbanken gespeichert werden. Dies ist bei multi-medialen Objekten wie etwa Bildern notwendig aber auch bei textbasierten Dokumenten kann die zentrale Speicherung der Metadaten günstig sein.

Was ist LMER?

Als Vertreter der Gruppe der Standards für technische Metadaten soll weiterhin der Standard für *Langzeitarchivierungsmetadaten für elektronische Ressourcen* (*LMER*) [25] erwähnt werden, der wie die meisten modernen Metadaten-Standards auf einem XML-Schema basiert. LMER wurde primär als Austauschformat konzipiert, die Speicherung wurde nicht berücksichtigt. Somit kann die Speicherung in beliebigen XML-Strukturen anderer Standards erfolgen. LMER deckt nicht den Anspruch auf Vollständigkeit ab. Besonders werden Metadaten, die von einem bestimmten Dateiformat abhängig sind, nicht berücksichtigt. Stattdessen sieht das XML-Schema an dieser Stelle vor, beliebige weitere XML-Strukturen zur formatabhängigen Beschreibung anzufügen. Diese Modularisierung ist ein großer Vorteil dieses Standards, um auch in Zukunft durch eine entsprechende Weiterentwicklung der Module auch den Ansprüchen neuer Dateiformate genügen zu können.

Wozu Premis (*Preservation Metadata: Implementation Strategies Working Group?*?
Einer der wichtigsten Vertreter eines Standards für generische Preservation Metadata ist PREMIS.

Das PREMIS-Datenmodell wurde entworfen, um die logischen Zusammenhänge zwischen den verwendeten Metadaten-Elementen (Entitäten) zu beschreiben.

Informationen zur Größe, Format und Beständigkeit können immer erfasst und gespeichert werden. Mit diesen generellen technischen Metadaten beschäftigt sich das PREMIS-Datenmodell. Andere Informationen fallen nur bei bestimmten Dateiformaten an und können

somit auch nicht grundsätzlich erfasst werden. Diese Informationen werden bei PREMIS nicht berücksichtigt.

Um Metadaten überhaupt mit dem zu archivierenden Objekt verknüpfen zu können, müssen diese Informationen gesammelt und auf strukturierte Art und Weise als komplexes Annotationsobjekt an das Dokument angeknüpft werden. Als de facto Standard für die Strukturierung und Übertragung der Metadaten zum Bestand hat sich das *METS-Datenmodell* etabliert.

Persistent Identifier

Unter dem Begriff eines *Persistenten Identifizierers* (engl. *Persistent Identifier*) ist eine Assoziation zwischen einem Namen und einer Ressource zu verstehen. Die Benennung von Objekten spielt insbesondere in der automatisierten Verarbeitung von digital kodierten Dokumenten und Medien-objekten eine grundlegende Rolle bezüglich deren Adressierung. Während Betriebssysteme in der Regel Dateinamen, Benutzer- und Prozess-Identitäten kennen, kommen in Netzwerken in der Regel Namens-Schemata wie etwa Ethernet-Adressen, IP-Adressen, DNS-Namen und URLs hinzu. In heterogenen Netzen, wird die Anzahl der Identifizierer-Arten noch größer. Prinzipiell wäre es also bei der Suche nach einem Persistenten Identifizierer möglich, sich einfach aus diesem Pool zu bedienen. Problematisch ist hierbei aber, dass die Internet-Umgebung vom ständigen Wandel gekennzeichnet ist. Neue Webserver nehmen täglich ihren Dienst auf, andere gehen - manchmal unbemerkt - vom Netz. Neue Formate und Anwendungen entstehen und alte geraten in Vergessenheit. Um einen einigermaßen verlässlichen, und vor allem langfristig gesicherten Zugriff auf digitale Daten im Internet zu erhalten, bedarf es deshalb zusätzlicher Konzepte und Methoden, die im weiteren Verlauf der Kurseinheit beschrieben werden. Vorweg jedoch noch eine wichtige Bemerkung hinsichtlich derer sich die Experten einig sind: Ein *Identifier-Schema* kann Persistenz immer höchstens unterstützen, niemals erzwingen, denn letztendlich ist Persistenz genauso ein technisches wie soziales Problem.

Anforderungen

Mit Blick auf Eigenschaften, die für einen Mechanismus zur Realisierung eines Persistenten Identifizierers von Vorteil wären, lassen sich folgende Kriterien benennen:

Eindeutigkeit: Ein Identifizierer sollte nie zwei verschiedene Objekte benennen können. Er stellt also eine (partielle) Funktion zwischen Namen und Objekten her.

Injektivität: Zwei verschiedene Identifizierer sollten verschiedene Objekte benennen.

Format das auf unbegrenzte Zeit hinaus unterstützt wird: Jedes Identifizierer-System ist nutzlos, wenn das Format für den Identifizierer irgendwann verloren geht.

Möglichst Technologie-unabhängig: Technologien sind bekanntlich schnellen Änderungen unterworfen. Es wäre daher naiv, zu glauben, dass z. B. ein Betriebssystem in 100 Jahren noch die gleiche Grundfunktionalität besitzt wie heute. Verlässt sich die Implementierung einer Identifizierer-Methode auf derartige Annahmen, stirbt die Methode mit der zugrunde liegenden Technologie aus.

Unabhängig vom Ort der Speicherung: Ist der Ort der Speicherung ein Teil des Identifizierers, wird er beim Ortswechsel ungültig (siehe z. B. DNS-Namen bei URL- Adressen)

Semantikfreiheit ist notwendig: Spiegelt ein Identifizierer wie auch immer geartete Semantiken z. B. die Besitzverhältnisse an der Ressource wieder, ist er bei Änderungen derselben (z. B. Verkauf) zumindest irreführend.

Interoperabilität des Formats: Bis ein uniformes allgemeingültiges Identifizierer-Format existiert, sollte das Format leicht andere Formate mit einschließen können bzw. von anderen eingeschlossen werden können.

Sprachunterstützung: Indentifizierer sollten mit Sprachen, die sich nicht leicht in ASCII darstellen lassen (z. B. chinesisch), zurechtkommen.

Einfachheit der Erstellung: Der Aufwand, um einen Identifizierer-Namen zu erstellen und einer Ressource zuzuordnen, sollte möglichst klein sein.

Einfachheit und Effizienz bei der Auflösung: Der Aufwand, um vom Namen zur Ressource, auf die der Identifier zeigt, zu kommen, sollte möglichst klein sein.
• **Ausfallsicherheit**: Falls der Server, der die zum Identifier gehörende Ressourcenzuordnung verwaltet, ausfällt, sollte ein anderer einspringen können, sonst entsteht der Eindruck, dass der Identifizierer-Name ungültig ist. Dies kann z. B. durch Redundanz gewährleistet werden.
• **Sicherheit:** Änderungen sollten nur durch autorisierte Personen möglich sein.
• **Meta-Information:** Es sollte möglich sein, Information über das referierte Objekt abzurufen.
• **Granularität:** Es wäre gut, wenn man die Zugriffsauflösung bzgl. der Resource abstufen könnte.

Allen Ansätzen zur Realisierung von Identifizierern ist gemeinsam, dass eine Indirektionsstufe eingeführt wird, sodass der Identifizierer vom Speicherungsort abgekoppelt wird.